(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3776302号
(P3776302)
(45)発行日　平成18年5月17日(2006.5.17)　(24)登録日　平成18年3月3日(2006.3.3)

(51)Int.Cl.　F I
G06F 9/38 (2006.01)　G06F 9/38 350A

請求項の数 3　(全 20 頁)

(21)出願番号　特願2000-310363(P2000-310363)
(22)出願日　平成12年10月11日(2000.10.11)
(65)公開番号　特開2001-117772(P2001-117772A)
(43)公開日　平成13年4月27日(2001.4.27)
審査請求日　平成16年10月7日(2004.10.7)
(31)優先権主張番号　09/417582
(32)優先日　平成11年10月14日(1999.10.14)
(33)優先権主張国　米国(US)

(73)特許権者　398038580
ヒューレット・パッカード・カンパニー
HEWLETT-PACKARD COMPANY
アメリカ合衆国カリフォルニア州パロアルト　ハノーバー・ストリート　3000
(74)代理人　100081721
弁理士　岡田　次生
(72)発明者　ドナルド・チャールズ・ソルティス・ジュニア
アメリカ合衆国80526コロラド州フォート・コリンズ、ローズゲート・コート　4414

最終頁に続く

(54)【発明の名称】コンピュータ・プログラムのハザードを検出するシステム

(57)【特許請求の範囲】

【請求項１】

コンピュータ・プログラムの命令を処理し、該コンピュータ・プログラム内のデータ・ハザードを検出するためのコンピュータ・システムであって、

複数のレジスタに対応して複数のエントリを有するメモリと、

複数のパイプラインと、

複数の命令と、該複数の命令からなるグループを識別する命令グループデータと、を有するコンパイルされたデータを受け取るよう構成された命令分散ユニット（ＩＤＵ）であって、該命令グループデータは、該命令グループにおける命令間にコンパイラがデータ・ハザードを検出したかどうかに基づくデータであり、さらに、該命令グループデータに基づいて、該コンパイルされたデータの前記命令を前記パイプラインに送るよう構成された命令分散ユニットと、

前記命令グループ内のそれぞれの命令を分析し、前記エントリのそれぞれを、対応するレジスタに関連づけられた命令が該命令グループ内で検出されたかどうかに基づいて制御し、該命令グループにおける該命令のうちの１つに関連づけられたレジスタに対応するエントリを識別し、該制御機構によって以前に分析された該命令グループのいずれかの命令が該レジスタに関連づけられているかどうかを、該エントリに基づいて判断し、該判断に基づいて、前記コンパイラによるグループ化のエラーを示す警告信号を送出するよう構成された制御機構と、

を備えるコンピュータ・システム。

【請求項２】
コンピュータ・プログラムの命令を処理し、該コンピュータ・プログラム内のデータ・ハザードを検出するためのスーパースカラー処理システムであって、
複数のレジスタに対応して複数のエントリを有するメモリと、
命令グループと、該命令グループにおける命令間にコンパイラがデータ・ハザードを検出しなかったことを示す命令グループデータと、を受け取り、該命令グループデータに基づいて、該命令を複数のパイプラインに送る手段と、
前記エントリのそれぞれを、対応するレジスタに関連づけられた命令が前記命令グループ内で検出されたかどうかに基づいて制御する手段と、
前記命令グループにおける第1の命令を分析する手段であって、
該第１の命令に基づいて、該第1の命令に関連づけられたレジスタに対応するエントリを前記メモリにおいて識別する手段と、
該エントリを分析する手段と、
前記命令グループ内の第２の命令が前記レジスタに関連づけられていることを前記エントリが示すかどうかを、前記命令グループデータに基づいて判断する手段と、
前記判断する手段による判断に基づいて、前記コンパイラによるグループ化のエラーを示す警告信号を送出する手段と、を有する分析する手段と、
を備える、スーパースカラー処理システム。
【請求項３】
コンピュータ・プログラム内のデータ・ハザードを検出する方法であって、
メモリの複数のエントリを、異なるレジスタのそれぞれに対応して保持するステップと、
複数の命令と、該複数の命令からなるグループを識別する命令グループデータと、を有するコンパイルされたデータを受け取るステップであって、該命令グループデータは、該命令グループにおける命令間にコンパイラがデータ・ハザードを検出したかどうかに基づくデータである、ステップと、
前記エントリのそれぞれを、対応するレジスタに関連づけられた命令が前記命令グループに含まれるかどうかに基づいて制御するステップと、
前記命令グループデータに基づいて、前記コンパイルされたデータの前記命令を、複数のパイプラインに送るステップと、
前記命令グループにおける第１の命令を分析するステップと、
前記第１の命令に関連づけられたレジスタに対応するエントリを、前記メモリにおいて識別するステップと、
前記識別するステップに応答して、前記エントリを分析するステップと、
前記エントリを分析するステップを介して、前記命令グループにおける第２の命令が前記レジスタに関連づけられていることを該エントリが示すかどうかを判断するステップと、
前記判断するステップの判断に基づいて、前記コンパイラによるグループ化のエラーを示す警告信号を送出するステップと、
を含む方法。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、一般的にはコンピュータ処理技術に関し、より具体的には、コンピュータ・プログラムの命令グループ内のデータ・ハザードを検出するスーパースカラー処理システムおよびその方法に関する。
【０００２】
【従来の技術】
スーパースカラー処理としても知られている並列処理は、コンピュータ・プログラムの命令を処理するのに必要な時間量を減少させるために開発されてきた。並列処理においては

、複数の命令を同時に実行する少なくとも２つのパイプラインが定義される。並列処理の１つのタイプは、アウトオブオーダー(out-of-order)処理である。アウトオブオーダー処理においては、プロセッサのそれぞれのパイプラインは、他のパイプラインからは独立して同時に異なる命令を実行する。

【０００３】

アウトオブオーダー処理においては、命令は、プロセッサによって受け取られる順序と同じ順序でパイプラインに入力されるとは限らない。更に、典型的には、命令が異なればそれを実行する時間量が異なるので、たとえある１つの命令がそのそれぞれのパイプラインに先に入力されていたとしても、他の命令が、当該１つの命令よりも前に実行を完了するということがあり得る。従って、命令は、プロセッサ内の複数パイプラインによって受け取られる順序と同じ順序で実行されるとは必ずしも限らない。その結果、アウトオブオーダー処理の場合、リード・アフター・ライト(read-after-write：書込み後の読み出し)のデータ・ハザードおよびライト・アフター・ライト(write-after-write：書込み後の書込み)のデータ・ハザードによって起こるエラーを防止する複雑さ(詳細は後述)が比較的大きい。

【０００４】

プロセッサによって実行されるべき１つの命令が、別の命令の実行によって抽出または生成されたデータを実行中に利用する時、"リードアフターライトのデータ依存性"が存在する。上記別の命令が実行する前に上記１つの命令が実行すると、実行中に不正データを利用することがあるので、エラーが発生することがある。結果として、エラーを防止するためには、上記別の命令の実行から必要なデータが利用可能となるまで、上記別の命令の実行によって抽出または生成されたデータを利用する上記１つの命令が実行しない、ということを保証するステップをとらなければならない。リードアフターライトのデータ依存性が存在し、かつそのようなステップがとられなければ、リードアフターライトのデータ依存性が不正データの利用となることがあり、よって"リードアフターライトのデータ・ハザード（data hazard）"が存在する。

【０００５】

一方、実行中に、相対的に若い命令によって書き込まれるレジスタまたはメモリ位置に、相対的に古い命令がデータを書き込み、該古い命令が、若い命令によって書き込まれた有効なデータを不正に上書きすることがある時、"ライトアフターライトのデータ・ハザード"が存在する。ここで、ある命令が、別の命令の後にプロセッサによって受け取られた時、その命令は、当該別の命令より"若い"という。反対に、ある命令が、別の命令より前にプロセッサによって受け取られた時、その命令は、当該別の命令より"古い"という。

【０００６】

ライトアフターライトのデータ・ハザードの１つの例として、第１の命令が、データを抽出して該抽出したデータを特定のレジスタに書き込むロード命令であると仮定する。抽出すべきデータがローカル的に利用可能でない場合、データを抽出するのに要する時間は比較的長い。従って、第１の命令より若い命令である第２の命令が、第１の命令が実行した後ではあるが該第１の命令によって抽出されたデータがレジスタに書き込まれるより前に、同じレジスタにデータを書き込む可能性がある。そのようなケースでは、第２の命令によってレジスタに書き込まれたデータが、第１の命令によって抽出されたデータによって上書きされることがある。その結果、レジスタは不正データを含み、後の命令がレジスタのデータを使用する時エラーが発生する可能性がある。

【０００７】

リードアフターライトのデータ・ハザードおよびライトアフターライトのデータ・ハザードによるエラーを防止するため、大部分のアウトオブオーダー並列プロセッサは制御機構を使用する。すなわち、制御機構は、それぞれの命令の実行中に、処理されつつある命令（以下、"ペンディング(pending)命令"と呼ぶ）が、より古い命令の実行によって生成されるデータを必要とするか否かを判断する。必要としていると判断すれば、制御機構は、次に、必要とされるデータが少なくとも利用可能となる時点までに当該古い命令の処理が

完了したか否かを判断する。このデータがなお利用可能でなければ、制御機構は、必要なデータが利用可能となるまでペンディング命令の処理をストール（stall、すなわち一時的に機能停止）し、こうしてリードアフターライトのデータ・ハザードによるエラーを防止する。

【０００８】

さらに制御機構は、古い命令からのデータ(すなわち古い命令によって抽出または生成されるデータ)が、ペンディング命令からのデータと同じレジスタまたは同じメモリ位置に書き込まれることになっているか否かを判断する。もしも同じレジスタまたは同じメモリ位置に書き込まれるならば、制御機構は、古い命令からのデータがそのレジスタまたはメモリ・アドレスに書き込まれるまでペンディング命令をストールし、それによって、ライトアフターライトのデータ・ハザードによるエラーを防止する。このようにして、制御機構は、リードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードによるエラーを防止するため、ペンディング命令をストールさせることができる。

【０００９】

ペンディング命令のストールは、通常、ペンディング命令を実行しているパイプラインに送られるストール信号をアサートすることによって達成される。アサートされたストール信号に応答して、パイプラインは、制御機構によってストール信号がデアサート（deassert）されるまでペンディング命令の実行を停止するよう設計される。リードアフターライトのデータ・ハザードもライトアフターライトのデータ・ハザードももはや存在しなくなれば、制御機構はストール信号をデアサートし、それに応じてパイプラインは、ペンディング命令の処理を再開する。リードアフターライトのデータ・ハザードおよびライトアフターライトのデータ・ハザードによる潜在的なエラーを検出して防止するのに必要とされる制御機構は、アウトオブオーダー・プロセッサにおいては比較的複雑であり、パイプラインの数が増加するにつれ、制御機構の複雑さは劇的に増加する。

【００１０】

この結果、多くの従来の並列プロセッサ、特に多数のパイプラインを有するプロセッサは、上述のアウトオブオーダーのタイプの処理方式の代わりにインオーダー(in-order)タイプの処理方式を使用する。インオーダー・タイプの処理方式においては、異なるパイプラインによって処理される命令は、システムクロック信号の所定のエッジに従ってパイプラインの複数の段階を１つずつ実行される。すなわち、１つのパイプラインにおける命令の処理は、通常複数の段階に分割され、パイプラインのそれぞれの段階は、異なる命令を同時に処理する。

【００１１】

例えば、それぞれのパイプラインによって実行される処理は、レジスタ段階、実行段階、例外検出段階および書込み段階に分割される。レジスタ段階の間、命令の実行に必要なオペランドが取得される。オペランドが取得されると、命令の処理は、命令が実行される実行段階に入る。命令が実行された後、命令の処理は例外検出段階に入り、そこで、例えばデータの信憑性が無いことを示すことがある実行中のオーバーランというような条件が検査される。例外検出段階が完了すると書込み段階に入り、そこで、実行段階の結果がレジスタに書き込まれる。

【００１２】

インオーダー処理の重要な特徴は、"発行グループ"のそれぞれの命令が、同時にそれぞれの段階を１つずつ実行する、という点にある。ここで、"発行グループ(issue group)"は、単一プロセッサ内で異なるパイプラインの同じ段階によって同時に(すなわち同一クロック・サイクルの間に)処理される命令セット、と定義される。例えば、従来技術において典型的に実施されているように、それぞれのパイプラインのそれぞれの段階が一度に１つの命令を処理すると仮定する。（複数の）パイプラインの例外検出段階における(複数の)命令が第１の発行グループを形成し、（複数の）パイプラインの実行段階における(複数の)命令が第２の発行グループを形成する。更に、（複数の）パイプラインのレジスタ

段階における（複数の）命令が第３の発行グループを形成する。ストールがないとすると、発行グループのそれぞれは、システム・クロック信号のアクティブ・エッジに応答して次のそれぞれの段階に進む。言い換えると、システム・クロック信号のアクティブ・エッジに応答して、第１の発行グループは書込み段階に進み、第２の発行グループは例外検出段階に進み、第３の発行グループは実行段階に進む。

【００１３】

"アクティブ・エッジ（active edge）"は、ここではシステムクロック信号の何らかのエッジを示すものとして使用されており、このエッジの発生により、パイプラインにおいてストールされていない命令のそれぞれは、そのパイプラインにおける次の処理段階に前進するよう誘導される。例えば、プロセッサが、ストールされていない命令を、３クロック毎に次の処理段階に進むよう設計されていると仮定する。この例では、アクティブ・エッジを、クロック信号の３番目毎の立ち上がりエッジとして定義することができる。どのクロック信号のエッジが"アクティブ・エッジ"として指定されるかは設計パラメータに基づいており、プロセッサごとに変えることができる、という点に注意すべきである。

【００１４】

インオーダー処理の間は、１つの発行グループにおけるいかなる命令も、別の発行グループの別の命令を追い越さない。言い換えると、上記別の発行グループの命令の後にパイプラインに入力された上記１つの発行グループの命令は、当該別の発行グループの命令のいずれかを処理する段階と同じ段階に入ることを禁止される。従って、いかなる時点においても、パイプラインのそれぞれの段階は、ただ１つの発行グループからの命令だけをそれぞれ処理している。異なる発行グループからの命令が相互に追い越すことが禁止されるので、パイプラインを制御して命令をストールし、リードアフターライトのデータ・ハザードおよびライトアフターライトのデータ・ハザードによるエラーを防ぐ制御機構が大幅に簡略化され、よってアウトオブオーダー処理に比べて好ましいことが多い。

【００１５】

しかしながら、インオーダーのプロセッサのなかには、リードアフターライトのデータ・ハザードおよびライトアフターライトのデータ・ハザードによるエラーが適切に防止されないものがある。この点に関して、プロセッサ・パイプラインによって処理される発行グループを定義するのに命令分散ユニット（ＩＤＵ：instruction dispersal unit）がたびたび利用される。更に、明示的並列命令今コンピューティング（ＥＰＩＣ：explicitly parallel instruction computing）を利用するプロセッサのようなインオーダー・プロセッサのなかには、命令が、命令グループの形でIDUに送られるものがある。命令グループは命令のセットであり、該セットの中の命令と命令の間にリードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードを持たないということを、プロセッサの外部のコンパイラまたは他の何らかの装置によって保証された命令セットである。

【００１６】

例えば、多くのコンパイラは、命令をIDUに順次送る。命令をコンパイルする際、コンパイラは、リードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードが存在するか否かをを判断する。パフォーマンスを最適化するため、コンパイラは、IDUが一層効率的に命令を処理することができるように、ストップ・ビットを挿入することによって命令グループを定義する。ここで使用される"ストップ・ビット"は、処理システムに送られる命令と命令の間に挿入されるビットであり、該ビットを適切にアサートまたはデアサートして、命令グループの開始および終了を示すことができる。

【００１７】

この点に関して、コンパイラは、連続的に送られる複数の命令が１つの命令グループを定義する時を判断し、グループの最初の命令の前およびグループの最後の命令の後にストップ・ビットをアサートするよう設計されることができる。その結果、アサートされたストップ・ビットとストップ・ビットの間の命令が１つの命令グループを定義し、よってアサートされたストップ・ビットとストップ・ビットの間のいかなる命令も、当該ストップ・

ビットとストップ・ビットの間の他の命令とリードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードを持つことがない、ということをIDUが認識することとなる。従って、IDUは、発行グループを定義する際に、命令グループ内の命令間におけるリードアフターライトのデータ・ハザードおよびライトアフターライトのデータ・ハザードについて検査する必要がない。

【００１８】

【発明が解決しようとする課題】

しかしながら、コンパイラが命令グループの中にリードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードを持つ２つの命令を間違って含む場合に問題が発生する。IDUは、同じ命令グループの中の命令間のリードアフターライトのデータ・ハザードおよびライトアフターライトのデータ・ハザードについて検査しないように設計されることがあるので(検査する代わりに、ストップ・ビットのアサート／デアサートを当てにして、リードアフターライトおよびライトアフターライトのデータ・ハザードを示すよう設計されることがある)、同じ発行グループにおける２つの命令がそれらの間にリードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードを持つ、というような発行グループを、IDUが不適切に定義することがある。同じ発行グループの２つの命令の間にリードアフターライトまたはライトアフターライトのデータ・ハザードを持つことは、インオーダー・プロセッサのアーキテクチャに違反し、その結果、プロセッサ・パイプラインによる２つの命令の処理中にエラーが発生する可能性が生じる。

【００１９】

このように、間にリードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードを持つ２つの命令を命令グループが含む場合を判断するシステムおよび方法を提供するという産業上の必要性が存在する。

【００２０】

【課題を解決するための手段】

本発明は、上述された従来技術の不適切な点および欠点を克服する。本発明は、一般的には、ある１つの命令グループが、同じ命令グループの他の命令とデータ・ハザードを規定する命令を含んでいるかどうかを判断するシステムおよび方法を提供する。

【００２１】

アーキテクチャの観点から見れば、本発明のシステムは、メモリ、複数のパイプライン、命令分散ユニット(IDU)、および制御機構を使用する。メモリは、複数のレジスタにそれぞれ対応する複数のエントリを有する。IDUは、複数の命令を含む命令グループを受け取り、該命令グループの命令を複数のパイプラインに送る。制御機構は、前記命令のうちの１つを分析し、前記１つの命令に関連するレジスタに対応するメモリのエントリを識別する。その後、制御機構は、エントリを分析して、前記命令グループ内の他の命令が前記レジスタに関連するということを前記エントリが示すという制御機構の判断に応答して、警告信号を送る。警告信号に応答して警告メッセージを生成し、ユーザに、命令グループの２つの命令がリードアフターライトまたはライトアフターライトのデータ・ハザードを規定するということを知らせることができ、および／または命令のさらなる処理を終了させることができる。

【００２２】

本発明の他の特徴によると、制御機構は、命令グループのそれぞれの命令が制御機構によって分析されたと判断した時、メモリにおけるエントリをリセットする。

【００２３】

本発明はまた、コンピュータ・プログラムの命令を処理し、該コンピュータ・プログラム内のハザードを検出するスーパースカラー処理方法を提供するものとみることができる。該方法は、複数の命令を含む命令グループを定義するステップと、該命令のうちの１つを分析するステップと、該１つの命令に関連するレジスタに対応するメモリ・エントリを識別するステップと、前記識別するステップに応答して前記エントリを分析するステップと

、前記分析するステップを介して、前記命令グループ内の他の命令が前記レジスタに関連するということを前記エントリが示すかどうかを判断するステップと、前記命令グループ内の前記他の命令が前記レジスタに関連するという前記判断ステップの判断に応答して、警告信号を送るステップと、によって広く要約されることができる。

【００２４】

本発明の他の特徴および有利な点は、以下の詳細な説明を図面と共に検証することにより、当業者には明らかとなるであろう。このような特徴および有利な点は、本発明の範囲内に含まれるものと意図され、また特許請求の範囲によって保護されるよう意図されている。

【００２５】

【発明の実施の形態】

本発明は、ある１つの命令グループが、同じ命令グループにおける他の命令とデータ・ハザードを持つ命令を含むかどうかを判断するスーパースカラー処理システムおよびその方法に関する。図１は、本発明の処理システム20を使用するコンピュータ・システム15の好ましい実施形態を示す。好ましい実施形態の処理システム20はハードウェアで実現されることが好ましいが、処理システム20の一部を必要に応じてソフトウェアで実現することも可能である。

【００２６】

図１に示されているように、コンピュータ・システム15は、１つまたは複数のバスを有することのできるローカル・インタフェース22を備える。ローカル・インタフェース22は、処理システム20がコンピュータ・システム15の他の構成要素と通信することを可能にする。さらに、キーボードおよび／またはマウスなどのような入力装置25を使用してシステム15のユーザからデータを入力することができ、ディスプレイ27および／またはプリンタ29を使用してユーザにデータを出力することができる。システム・クロック31はクロック信号を生成し、該クロック信号は、当該技術分野で既知の技術を介して、システム15によって伝達されるデータのタイミングを制御するのに使用される。ディスク記憶機構32をローカル・インタフェース22に接続して、(たとえば磁気または光学的な)不揮発性ディスクとの間でデータを転送することができる。また必要に応じて、システム15をネットワーク・インタフェース33に接続して、ネットワーク35とデータを交換することを可能にすることができる。

【００２７】

さらに、システム15は、メモリ44に記憶されるプログラム41、システム・マネージャ42およびコンパイラ46を備える。プログラム41は、処理システム20によって処理および実行される命令を有する。システム・マネージャ42は、入力装置25および／またはネットワーク・インタフェース33から入力を受け取り、必要に応じてプログラム41の命令を処理システム20に送るよう設計される。プログラム41の命令は、処理システム20に送られる前に、好ましくは先ず処理システム20と互換性のある形式にコンパイラ46によって変換される。例えば、プログラム41の命令が(CまたはFortranなどのような)高水準コンピュータ言語で書かれている場合、コンパイラ46は、処理システム20と互換性のあるマシン語にそれらの命令を翻訳するよう設計される。

【００２８】

本発明の好ましい実施形態において、コンパイラ46は、翻訳された命令を含み、かつ処理システム20に直接送られることのできる命令バンドル（instruction bundle）を定義する。図２は、本発明の原理に従う命令バンドル52を示している。図２に示されているように、それぞれのバンドル52は、１つまたは複数の命令を定義するデータを含み、さらにヘッダ55を有する。ヘッダ55は、バンドル52に含まれる命令の種類を特定する識別子情報を有する。例えば、ヘッダ55は、バンドル52における第１の命令がメモリ演算命令であり、バンドル52における第２の命令が整数演算命令であり、バンドル52における第３の命令が浮動小数点演算であることを示す情報を有することができる。ヘッダ55はまた、ストップ・ビット57を有する(これについては、詳細を後述する)。図２に示される命令バンドル52は

３つの命令を含むよう示されているが、任意の数の命令をバンドル52に含めることができる。

【００２９】

命令バンドル52を定義する際、コンパイラ46は、リードアフターライトのデータ・ハザードおよびライトアフターライトのデータ・ハザードについて検査し、リードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードを規定する２つの命令が同じバンドル52内に置かれないということを確実にするよう設計されるのが好ましい。加えて、コンパイラ46は、バンドル52をプログラムの順に(すなわち、命令が実行される順に)処理システム20に順次送るよう設計される。さらにコンパイラ46は、アサートされたストップ・ビット52によって区切られた複数のバンドル52における命令と命令の間に、いかなるリードアフターライトのデータ・ハザードおよびライトアフターライト・データ・ハザードも存在しないことを確実にするよう設計されるのが好ましい。従って、処理システム20は、デアサートされたストップ・ビット57を持つ一連の命令バンドル52を受け取った場合には、該一連の命令バンドルにおける命令のいずれもが、該一連の命令バンドルにおける他のいかなる命令にも依存しないこと、または、該一連の命令バンドルにおける命令のいずれもが、該一連の命令バンドルの他の命令と同じレジスタにデータを書き込まないこと、に気づく。

【００３０】

例えば、図３を参照すると、図３によって示される順序で処理システム20に順次送られるバンドル52a～52gというストリングが示されている。図３において、"D"を有するそれぞれのバンドル52a、52c、52dおよび52eは、デアサートされたストップ・ビットを持ち、"A"を有するそれぞれのバンドル52b、52fおよび52gは、アサートされたストップ・ビットを持つ。ストップ・ビット57を分析することにより、コンパイラ46によればバンドル52f内の命令とバンドル52g内の命令との間にリードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードが存在する可能性がある、ということを判断することができる。しかしながら、バンドル52c、52dおよび52eはデアサートされたストップ・ビット57を持つので、コンパイラ46によれば、バンドル52c、52d、52eおよび52fのいずれの命令の間においてもリードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードが存在しない、ということを判断することができる。言い換えると、バンドル52c、52d、52eおよび52fは１つの命令グループを定義する。さらに、バンドル52bはアサートされたストップ・ビット57を持つので、バンドル52aおよび／または52bは、バンドル52c、52d、52eおよび52fによって定義される命令グループ内の命令の１つとリードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードを持つ命令を含む可能性がある。

【００３１】

本発明の原理から逸脱することなく、処理システム20に命令バンドル52を送る他の方法を使用することができる、ということに留意すべきである。処理システム20にバンドル52を送り、かつ命令グループの存在を示す方法ならば、どのような方法も本発明を実現するのに適している。

【００３２】

図４に示されているように、処理システム20は、処理システム20に送られた命令バンドル52を受け取るよう設計される命令分散ユニット(IDU)72を含む。IDU72は、IDU72によって受け取られた命令バンドル52の命令を使用して発行グループを定義して、１つの発行グループの命令をパイプライン75に送るよう構成されており、発行グループのそれぞれの命令が、パイプライン75のうちのただ１つによってのみ受け取られるようにする。パイプライン75は、受け取った命令を更に処理および実行するよう設計される。並列インオーダープロセッサの従来のパイプラインと同様に、パイプライン75は、受け取った命令を複数の段階で処理するのが好ましい。

【００３３】

図５は、パイプライン75についての典型的な一組の段階を示す。すなわち、パイプライン

75のそれぞれは、レジスタ段階77、実行段階79、例外検出段階81および最後に書込み段階83において、命令を受け取って該命令を順次処理する。これらの段階についての詳細は上記の従来技術の項に記述したが、命令を処理および実行するために、他の段階および／またはいくつかの段階の他の組み合わせを使用することができる、ということは注意すべきであろう。

【００３４】

発行グループを定義する際、IDU72は、それぞれの命令が、その命令と互換性を持つパイプライン75にのみ送られるということを保証するよう設計されるのが好ましい。すなわち、パイプライン75の中には、所定の種類の命令だけを取り扱うように設計されている場合がある。例えば、パイプライン75の１つまたは複数を、メモリ演算命令、整数演算命令、浮動小数点命令または他の既知の種類の命令だけを取り扱うよう構成することができる。従って、IDU72は、受け取った命令を分析して、適切な種類の命令がそれぞれのパイプライン75に送られるように発行グループを定義するよう設計される。好ましい実施形態において、IDU72は、それぞれの命令バンドル52のヘッダ55を分析して、どの命令がどのパイプライン75と互換性があるかを判断することができる。

【００３５】

IDU72はまた、リードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードを規定する２つの命令が、同じ発行グループに置かれない、ということを確実にするよう設計される。従って、クロック31によって生成されるクロック信号のアクティブ・エッジ上で処理の第１の段階(すなわち、好ましい実施形態においてはレジスタ段階)に入るそれぞれの命令は、同じクロックエッジ上で第１の段階に入る他の命令のいずれに対してもデータ依存性を持ってはならない。更に、クロック信号のエッジ上で処理の第１の段階に入るそれぞれの命令は、同じクロックエッジ上で第１の段階に入る他のどの命令とも同じレジスタにデータを書き込んではならない。

【００３６】

上述のように、命令バンドル52のストップ・ビット57は、リードアフターライトのデータ依存性またはライトアフターライトの・データ・ハザードが、連続する命令バンドル52の命令と命令の間に存在するかどうかを示すので、IDU72は、ストップ・ビット57を利用して、発行グループを定義するプロセスを単純化するのが好ましい。すなわち、IDU72は、命令グループの中の命令と命令の間のリードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードを検査することなく、１つの命令グループを定義する一連のバンドル52における命令のうち任意の命令を、同じ発行グループに含めることができる。

【００３７】

IDU72はさらに、古い命令よりも先により若い命令がパイプライン75の処理を完了しないことを保証するよう設計される。この点に関して、周知のことではあるが、命令の処理は、元のプログラム41によって定義されたものと同じ順序で(すなわち、"プログラム順序"で)完了されなければならない。このプログラム順序は、命令が処理システム20に送られる順番である。

【００３８】

それぞれの命令の新旧は、"プログラム順序"におけるその位置に基づく。例えば、プログラム41において実行されるべき第１の命令(すなわち、処理システム20によって受け取られるプログラムの第１の命令)は最も古い命令であり、そのプログラムの他のすべての命令はこの命令より若い。第１の命令の後に実行されるべき次の命令（すなわち、第１の命令の後に処理システム20によって受け取られるプログラムの次の命令）は、第１の命令より若いが、プログラム41の残りの命令より古い。さらに、実行されるべき最後の命令は、最も若い命令である。たとえスーパースカラー・プロセッサが一度に複数の命令を処理するとしても、それらの命令は、非スーパースカラー・プロセッサがプログラム41を１ステップずつ実行して一度に一つずつ命令を処理している場合と同じ順序で処理を完了しなければならない（すなわち、この例では書き込み段階83を完了しなければならない）。より

若い命令がより古い命令よりも先に処理を完了しない、ということを確実にするため、IDU72は、より古い命令を、より若い命令を含む発行グループの後にパイプライン75に送られる発行グループには割り当てないのが好ましい。

【００３９】

IDU72は、発行グループが定義されると、インオーダー方式でそれぞれの発行グループをパイプライン75に順次送るよう設計される。従って、発行グループ内のそれぞれの命令は、クロック信号の同じアクティブ・エッジ上で、そのそれぞれのパイプライン75に送られる。理想的には、それぞれの発行グループ内のそれぞれの命令は、クロック信号のアクティブ・エッジとアクティブ・エッジとの間のそのそれぞれの段階で完全に処理され、発行グループ内のそれぞれの命令が同じクロックエッジ上で次の段階に進むようにする。従って、ストールが無い場合、レジスタ段階77における発行グループの命令は、実行段階79および例外検出段階81における発行グループの命令が例外検出段階81および書込み段階83にそれぞれ入るのと同じクロックエッジ上で、実行段階79に入る。さらに、レジスタ段階77、実行段階79および例外検出段階81における発行グループの命令が次のそれぞれの段階に進むとき、新しい発行グループの命令がレジスタ段階77に入る。結果として、発行グループの処理は、ある発行グループからのいかなる命令も、他の発行グループ内の命令と同じ段階に入ることがないよう制御される。

【００４０】

図４に示されているように、処理システム20は、パイプライン75に接続された制御機構85を備えるのが好ましい。単純化のため、図４の制御機構85は、パイプライン75のうちの１つにだけに接続しているように示されているが、好ましい実施形態においては、制御機構85は、パイプライン75のそれぞれに同じように接続される。

【００４１】

制御機構85は、リードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードによるエラーを防止するため、パイプライン75によって処理されるデータを分析して必要に応じて命令をストールするよう設計される。すなわち制御機構85は、リードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードが、異なる発行グループの命令と命令の間に存在するかどうかを判断するよう構成される。その後、制御機構85は、上記ハザードのうちの１つに起因するエラーを引き起こす可能性を持つ命令または該命令を含む発行グループを、少なくともエラーを起こす可能性がなくなるまで、ストールさせるよう構成される。係属中の米国特許出願"Superscalar Processing System and Method for Efficiently Preventing Errors Caused by Write-After-Write Data Hazards"」（docket no. 10971195）、および米国特許出願"Superscalar Processing System and Method for Efficiently Performing In-Order Processing of Instructions"（docket no. 0971338）は、両方ともこの発明の発明者によって出願されたものであり、ここで参照により取り入れる。これらの出願は、パイプライン75によって処理されている命令をストールさせるシステムおよび方法を開示している。

【００４２】

しかしながら、従来の制御機構は、一般に、ハザードを発生させる命令が同じ発行グループに置かれている場合、リードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードによるエラーを防止するようには設計されていない。更に、上述の従来技術の項で記述したように、コンパイラ46がリードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードを規定する２つの命令が同じ命令グループにあるということを間違って示した場合、IDU72は、該２つの命令を間違って該同じ発行グループに割り当てる可能性がある。従って、コンパイラ46がリードアフターライトのデータ・ハザードまたはライトアフターライトのデータ・ハザードを規定する２つの命令を同じ命令グループ内に間違って含めてしまった場合を知ることが望ましい。この情報は、この２つの命令が同じ発行グループに割り当てられた場合にエラーを検出するのに役立つものであり、また、たとえこの２つの命令が現在同じ発行グループに含まれていないとしても将来発生するかもしれない潜在的なエラーを検出するのにも役立つ。

【００４３】
この点に関して、IDU72はこの２つの命令を異なる発行グループに割り当てることができ、制御機構85は、該２つの命令のうちの１つをストールさせることによっていかなるエラーをも防止することができる。しかしながら、この２つの命令が、プログラム41の将来の実行においても異なる発行グループに割り当てられるという保証はない。従って、たとえ２つの命令がIDU72によって同じ発行グループに現在は割り当てられていないとしても、該２つの命令が同じ命令グループに含められる場合を知ることが望ましい。従って、パイプライン75によって処理されているデータを分析する際、制御機構85は、命令グループのいずれかの命令と同じ命令グループの他のいずれかの命令との間にリードアフターライトのデータ依存性またはライトアフターライトのデータ・ハザードが存在するかどうかを判断するのが好ましい。

【００４４】
好ましい実施形態において、IDU72は、命令グループの少なくとも１つの命令にデータを挿入し(すなわち、少なくとも１つの命令にタグを付け)、制御機構85が、プログラム順序で命令を分析する時にどの命令が同じ命令グループに関連づけられているかを判断することができるようにする。例えば、命令がプログラム順序で分析されているときに、命令グループの最初または最後の命令にタグを付けて、命令グループの開始または終了をそれぞれ示すことができる。しかし、本発明の原理から逸脱することなく、どの命令がどの命令グループに関連しているかを制御機構85に示す他の方法を使用することもできる。

【００４５】
図４に示されているように、制御機構85はメモリ91を備える。メモリ91は、処理システム20の１つのレジスタにつき１つのエントリを含むのが好ましい。この点について、処理システム20は、従来技術の処理システムと同様に、パイプライン75による命令処理の間にデータを読み出すまたはデータを書き込むことのできる所定数のレジスタを有する。したがって、メモリ91はデータ・ビット列を有し、ここでそれぞれのビットは、処理システム20のレジスタのうちの１つを表す。言い換えると、それぞれのビットは、処理システム20の特定のレジスタに対応する、メモリ91のエントリである。このビット列を、メモリ91の単一アドレスに記憶することもでき、またはメモリ91の複数アドレスに記憶することもできる。他の例では、それぞれのエントリは、メモリ91の異なるアドレスに置かれる。すなわち、メモリ91の特定のアドレスに記憶されたデータは、処理システム20の特定のレジスタを表す。結果として、特定のレジスタに対応するデータを、データのアドレスに基づいて識別することができる。

【００４６】
例示の目的から、以降、メモリ91のエントリは、メモリ内の特定のアドレスに置かれたビット列内のビットに対応し、該ビット列のそれぞれのビットが、エントリ、したがって処理システム20の特定のレジスタに対応すると仮定する。しかしながら、本発明の原理から逸脱することなく、処理システム20のそれぞれのレジスタをメモリ91のエントリで表すのに様々な他の方法が存在し、これらの方法を他の実施形態で使用することができる。さらに、図４に示されるように、この好ましい実施形態ではメモリ91は制御機構85内に置かれているけれども、必要に応じて、メモリ91をコンピュータ・システム15の外部または内部の他の位置に置くことができる。

【００４７】
従来のインオーダー処理システムと同様に、制御機構85は、発行グループが段階77、79、81または83のうちの１つによって処理されている間に、該発行グループのそれぞれの命令を分析して、これらの命令のいずれかをストールすべきかどうかを判断するよう設計される。制御機構85は、書込み命令を分析する時、該書込み命令によって書き込まれるレジスタに対応するメモリ91のエントリを分析するよう設計される。書込み命令は、処理システム20に関連するレジスタにデータを記憶する任意の命令である。

【００４８】
上記のエントリは、該エントリに関連するレジスタにデータを書き込む同じ命令グループ

内の他の書き込み命令が制御機構85によって前に検出されたかどうかを示すのが好ましい。好ましい実施形態では、そのような他の命令が検出されたならば、エントリを定義するビットがアサートされ、そのような他の命令が検出されなかったならば、デアサートされる。したがって、ビットがアサートされているならば、制御機構85は、同じ命令グループ内の他の書き込み命令が、現在分析されている命令によって使用されるレジスタと同じレジスタを使用する、ということに気づく。言い換えると、制御機構85は、ライトアフターライトのデータ・ハザードが存在するということに気づく。

【００４９】

このような判断に応答して、制御機構85は、現在分析されている命令が、同じ命令グループ内の他の命令によって書き込まれるレジスタと同じレジスタを使用するということを示す警告信号を送るよう設計される。しかしながらビットがデアサートされている場合には、制御機構85は該ビットをアサートし、書き込み命令がエントリに対応するレジスタを使用するということを該エントリが示すようにする。ビット値を論理的に高(high)または低(low)のいずれかで記憶することにより、アサートまたはデアサートすることができるということは周知のことである。

【００５０】

さらに、制御機構85は、読み出し命令を分析する時、読み出し命令によって読み出されるレジスタに対応するメモリ91のエントリを分析するよう設計される。読み出し命令は、処理システム20に関連するレジスタからデータを読み出す任意の命令である。エントリのビットがアサートされていれば、制御機構85は、当該読み出し命令によって使用されるレジスタと同じレジスタにデータを書き込む同じ命令グループ内の書き込み命令を前に分析したということに気づく。言い換えると、制御機構85は、リードアフターライトのデータ・ハザードが存在するということに気づく。従って、ビットがアサートされているという判断に応答して、制御機構85は、該制御機構85によって分析されている命令が、同じ命令グループ内の他の命令によって書き込まれるレジスタと同じレジスタを使用する、ということを示す警告信号を送るよう構成される。

【００５１】

さらに、制御機構85は、命令グループ内の命令のそれぞれが分析されたと判断したとき、メモリ91のエントリをリセットするよう設計される。エントリをリセットした後、それぞれのエントリは、該エントリに対応するレジスタを使用する前の命令が検出されていないことを示す。好ましい実施形態では、制御機構85は、メモリ91のエントリを定義するそれぞれのビットをデアサートすることによってエントリをリセットする。

【００５２】

制御機構85は、プログラム順序で命令を分析するので、１つの命令グループのすべての命令を、他の命令グループの命令のいずれかを分析する前に完全に分析する。従って、制御機構85は、１つの命令グループの最後の命令を分析した後であって（さらに必要に応じて、該最後の命令によって使用されるレジスタに対応するエントリのビットをアサートした後）、次の命令グループの最初の命令を分析する際にメモリ91を分析する前に、メモリ91のエントリのビットをデアサート（すなわち、リセット）することができる。この機能を達成する種々の方法を使用することができるけれども、制御機構85は、分析されている命令が命令グループの最初の命令または最後の命令であるという判断に応答して、メモリ91のビットをリセットすることができる。制御機構85は、上述のようにIDU72によって命令グループの最初または最後の命令に挿入されるデータを分析することによって、この判断を行うことができる。

【００５３】

好ましい実施形態では、制御機構85によって送られた警告信号はシステム・マネージャ42によって受け取られ、システム・マネージャ42は、該警告信号に応答してプログラム41の実行を終了させることができる。さらに、警告信号はディスプレイ27および／またはプリンタ29によって受け取られることができ、ディスプレイ27および／またはプリンタ29は、違反が発生したことをユーザに示す警告メッセージを生成することができる。警告信号お

よび警告メッセージは、制御機構85が違反を発見した時どの命令が分析されていたかをユーザが判断することができるよう十分な情報を含むのが好ましい。

【００５４】

留意されるべき点であるが、読み出し命令がパイプライン75に入力されると、大部分の読み出し命令により、データがそのレジスタが比較的迅速に読み出される。一方、大部分の書込み命令により、データは、さらにパイプライン75の処理を進んだ後にそのレジスタに書き込まれる。例えば、読み出し命令によりデータは典型的にはレジスタ段階77においてその関連するレジスタから読み出され、書込み命令によりデータは典型的には書込み段階83においてその関連するレジスタに書き込まれる。従って、若い書込み命令をもつ同じ命令グループの読み出し命令は、その書込み命令がデータをレジスタに書き込む前にそのレジスタからデータを読み出す可能性が高い。従って、読み出し命令が書込み命令より古い時、たとえ読み出し命令および書込み命令が同じ命令グループにあるとしても、エラーが発生する可能性は少ない。結果として、若い書込み命令と同じレジスタを利用する古い読み出し命令が同じ命令グループにある時、ユーザが警告されることはそう重要なことではない。言い換えると、ライトアフターリードのデータ・ハザードに関して警告信号を生成することは、そう重要なことではない。

【００５５】

したがって、好ましい実施形態では、制御機構85は、読み出し命令を分析する時、該読み出し命令によって使用されるレジスタに対応するエントリのビットをアサートすることはしない。代わりに、書き込み命令にのみ応答して、メモリ91のビットはアサートされる。結果として、読み出し命令が、同じグループのより若い読み出し命令または書き込み命令と同じレジスタを使用する場合には、警告信号は生成されず、よってユーザーは警告されない。

【００５６】

しかしながら、必要に応じて、ユーザーは、読み出し命令がより若い読み出し命令と同じレジスタを使用する場合に警告を受けることもできる。言い換えると、同じグループ内の２つの命令がライトアフターリードのデータ・ハザードを規定することを制御機構85が検出した時はいつでも、警告信号を生成することができる。この機能を達成するため、制御機構85は、メモリ91内のエントリのデータを、該エントリに関連するレジスタを使用する読み出し命令を制御機構85が前に検出したかどうかを示すよう設定する。例えば、それぞれのエントリがデータ・ビットである実施形態では、制御機構85は、読み出し命令によって使用されるレジスタに対応するエントリのビットをアサートするよう構成されることができる。従って、書き込み命令が制御機構85によって前に検出されたかどうかを示す代わりに、エントリは、読み出し命令が前に検出されたかどうかを示す。

【００５７】

制御機構85が書き込み命令を分析するとき、制御機構85は、該書き込み命令によって使用されるレジスタに対応するエントリを検査する。エントリが、古い読み出し命令が検出されたことを示しているならば（前述した例では、エントリのビットがアサートされているならば）、制御機構85は、命令グループ内にライトアフターリードのデータ・ハザードが存在するということに気づき、警告信号を送って、命令グループ内にライトアフターリードのデータ・ハザードが存在することを示す。

【００５８】

それぞれのレジスタに対応するエントリを２つ持つことにより、制御機構85は、リードアフターライトおよびライトアフターライトのデータ・ハザードと同様に、ライトアフターリードのデータ・ハザードを検出することができる。この点について、一方のエントリを使用して、対応するレジスタを使用する古い書き込み命令が検出されたかどうかを示すことができ、他方のエントリを使用して、対応するレジスタを使用する古い読み出し命令が検出されたかどうかを示すことができる。また、前の命令がレジスタを使用したかどうかを示すだけでなく、何の種類の命令（すなわち、書き込みおよび／または読み出し）が前にそのレジスタを使用したかを示す単一のエントリを定義することも可能である。

【００５９】
更に留意されるべき点であるが、本発明はインオーダー処理を背景として記述されている。しかし、本発明の原理を、アウトオブオーダー処理に適用することができる。すなわち、アウトオブオーダー・プロセッサも、上述したような命令グループを介して実行されるべき命令を受け取ることができる。従って、アウトオブオーダー・プロセッサが制御機構85と同様の機構であって、それぞれの命令グループの命令によってどのレジスタが使用されるかを追跡する機構を含む限り、本発明の原理をアウトオブオーダー・プロセッサについても実現することができる。
【００６０】
動作
処理システム20および関連する方法の好ましい使用および動作を以下に記述する。発行グループがパイプライン75を順次進むにつれて、制御機構85は、段階77、79、81および83を通過する命令を分析する。段階77、79、81または83のうちの１つにおいて命令を分析する時、制御機構85は、図６に示される機能を実行する。以下の記述において、例示の目的から、制御機構85が実行段階79において命令を分析する際に図６の機能を実施すると仮定する。しかし当然のことながら、制御機構85がその他の段階77、81または83のいずれかにおいて命令を分析する際に図６の機能を実施することは可能である。
【００６１】
図６のブロック108に示されるように、最初に制御機構85は、段階79の最初の命令を分析し、次にブロック112において、その命令が新しい命令グループに関連するかどうかを判断する。命令が新しい命令グループに関連するならば、制御機構85は、ブロック114においてメモリ91のエントリをリセットし、それぞれのエントリが、該エントリに対応するレジスタが前の命令によって使用されていないことを示すようにする。好ましい実施形態においてこの機能を達成するため、制御機構85は、レジスタに関連するメモリ91のそれぞれのビットをデアサートする。
【００６２】
ブロック112を実行した後、または該当する場合にはブロック114を実行した後、ブロック115において、制御機構85は、該制御機構85によって分析されている命令（以下、「現命令」という）によって使用されるレジスタに対応するメモリ91のエントリを分析する。ブロック117および119に示されるように、命令グループ内の前に分析された書き込み命令が現命令と同じレジスタを使用することをエントリが示すならば、制御機構85は警告信号を送る。
【００６３】
好ましい実施形態において、レジスタに対応するエントリにビットがアサートされているとき、エントリは、命令グループ内の前に分析された書き込み命令が現命令と同じレジスタを使用するということを示す。ビットがデアサートされていれば、制御機構85は、現命令によって使用されるレジスタと同じレジスタに書き込む同じグループ内のいかなる他の命令も存在せず、よっていかなる警告信号も送られる必要がない、ということに気づく。結果として、制御機構85は、現命令によって使用されるレジスタに対応するビットがアサートされているという判断に応答して、ブロック119において警告信号を送る。
【００６４】
前に分析された書き込み命令が同じレジスタを使用するということをエントリが示さなければ、制御機構85は、ブロック123において、現命令が書き込み命令であるかどうかを判断する。書き込み命令であれば、制御機構85は、ブロック125において、現命令によって使用されるレジスタ（すなわち、現命令によって書き込まれるレジスタ）に対応するエントリにデータを設定することにより、命令グループの命令が、対応するレジスタにデータを書き込むということをエントリが示すようにする。従って、好ましい実施形態では、制御機構85は、現命令によって使用されるレジスタに対応するエントリのビットをアサートする。
【００６５】

ブロック119またはブロック123を実行した後、または該当する場合にはブロック125を実行した後、制御機構85は、ブロック128において、まだ分析されていない命令が段階79にあるかどうかを判断する。命令が残っていれば、制御機構85は、ブロック131において次の命令を分析し、該次の命令についてブロック112から開始する前述のプロセスを繰り返す。段階79における命令のすべてが分析されたならば、制御機構85は、ブロック135によって示されるように、上記のプロセス（ステップ108から開始するプロセス）を繰り返す前に、次の発行グループが段階79に到来するのを待つ。

【００６６】

前述のように、制御機構85を、ライトアフターリードのデータ・ハザードを検出するよう構成することができる。この機能を実行するため、制御機構85は、図７の機能を実現するよう構成される。図６を図７と比較することによってわかるように、図７によって示される機能は、図６のブロック117、123および125がブロック141、143および145によって置き換えられる点を除き、図６によって示される機能と同様である。さらに、ブロック149が追加される。

【００６７】

この点について、制御機構85は、図６のブロック117において、同じ命令グループにあり、かつ現命令と同じレジスタを使用する書き込み命令を前に検出したかどうかをブロック115を介して判断したが、図７のブロック141においては、制御機構85は、同じ命令グループにあり、かつ現命令と同じレジスタを使用する読み出し命令を前に検出したかどうかをブロック115を介して判断する。さらに、制御機構85は、図６のブロック123において現命令が書き込み命令かどうかを判断したが、図７のブロック143では、現命令が読み出し命令かどうかを判断する。さらに、制御機構85は、図６のブロック125において、現命令によって使用されるレジスタに対応するエントリのデータを、該レジスタを使用する書き込み命令が検出されたことを示すよう設定したが、図７のブロック145においては、現命令によって使用されるレジスタに対応するエントリのデータを、該レジスタを使用する読み出し命令が検出されたことを示すよう設定される。

【００６８】

図７のブロック149により、古い読み出し命令が他の読み出し命令と同じレジスタを使用するという判断に応答して、警告信号が送られないことが保証される。当該技術分野で知られているように、同じレジスタを使用する２つの読み出し命令は、たとえ該２つの命令が同じ命令グループにあるとしてもエラーを引き起こす可能性が少なく、通常ユーザーは、そのような事象を警告される必要がない。上記の違い以外については、図７の機能は図６の機能と同じである。結果として、警告信号は、同じ命令グループ内に古い読み出し命令および若い書き込み命令を制御機構85が検出した時に、制御機構85によって生成される。

【００６９】

コンピュータ・プログラム41の命令を分析するに際して、制御機構85を、図６および図７の両方の機能を実現するよう構成することができる。しかしながら、制御機構85は、ブロック117および141を実現するためにブロック115において２以上のエントリを分析する必要がある場合がある。この点について、特定のレジスタに対応する第１のエントリは、該特定のレジスタを使用する古い書き込み命令が現命令と同じ命令グループに存在するかどうかを示し、また第２のエントリは、上記特定のレジスタを使用する古い読み出し命令が現命令と同じ命令グループに存在するかどうかを示す。従って、制御機構85は、ブロック125において第１のエントリのデータを操作し、ブロック145において第２のエントリを操作することとなる。代わりに、エントリが、該エントリに対応するレジスタが命令グループの他の命令によって前に使用されたかどうかを示し、さらに該命令グループ内の何の種類（すなわち、読み出しおよび／または書き込み）の命令が、該対応するレジスタを使用したかをも示すことができる限り、それぞれのレジスタについて１つのエントリを使用することができる。

【００７０】

留意されるべき点であるが、制御機構85は、図６および図７によって示されている機能に加えて他の機能を実行することもできる。例えば、ブロック135において次の発行グループを待つ間、制御機構85は、他の段階77、81および／または83の他の命令を分析して、いずれかの命令をストールする必要があるかどうかを判断することができる。従って、当業者に明らかなように、制御機構85の機能は、図６および図７によって示される機能に限定されるべきではない。

【００７１】

本発明の上記の実施形態、特に"好ましい"実施形態は、実施形態の単なる可能な例に過ぎず、本発明の原理の明確な理解のために提示されているに過ぎない点は特に強調されるべきであろう。本発明の原理および理念を逸脱することなく上記実施形態に多くの変更および修正を加えることが可能である。

【００７２】

本発明には、例として次のような実施様態が含まれる。

（１）コンピュータ・プログラム(41)の命令を処理し、該コンピュータ・プログラム内のハザードを検出するシステム(20)であって、

複数のレジスタにそれぞれ対応する複数のエントリを有するメモリ(91)と、

複数のパイプライン(75)と、

複数の命令を含む命令グループを受け取り、該複数の命令を前記複数のパイプラインに送るよう構成された命令分散ユニット（ＩＤＵ）と、

前記複数の命令のうち少なくとも１つの命令を分析し、該１つの命令に関連するレジスタに対応する前記メモリのエントリを識別し、該エントリを分析し、前記命令グループ内の他の命令が前記レジスタに関連することを前記エントリが示すという判断に応答して警告信号を送るよう構成された制御機構(85)と、

を備えるコンピュータ・プログラム内のハザードを検出するシステム。

【００７３】

（２）前記制御機構が、前記命令グループの前記命令のそれぞれが該制御機構によって分析されたという判断に応答して、前記エントリをリセットするよう構成された上記（１）に記載のコンピュータ・プログラム内のハザードを検出するシステム。

【００７４】

（３）前記制御機構が、プログラム順に前記命令を分析するよう構成された上記（２）に記載のコンピュータ・プログラム内のハザードを検出するシステム。

【００７５】

（４）前記制御機構が、前記他の命令を分析し、該他の命令が前記レジスタに関連するという判断に応答して前記エントリにデータ値を記憶するよう構成された上記（１）に記載のコンピュータ・プログラム内のハザードを検出するシステム。

【００７６】

（５）前記制御機構が、前記他の命令が書き込み命令であるという判断に応答して、前記エントリに前記データ値を記憶するよう構成された上記（４）に記載のコンピュータ・プログラム内のハザードを検出するシステム。

【００７７】

（６）コンピュータ・プログラム(41)の命令を処理し、該コンピュータ・プログラム内のハザードを検出するスーパースカラー処理方法であって、

複数の命令を含む命令グループを定義するステップと、

前記命令のうちの１つを分析するステップと、

前記１つの命令に関連するレジスタに対応するメモリ・エントリを識別するステップと、

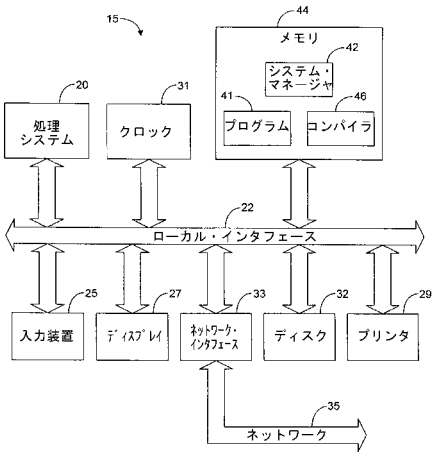
前記エントリを識別するステップを介して、前記命令グループ内の他の命令が前記レジスタに関連することを前記エントリが示すかどうか判断するステップと、

前記命令グループ内の他の命令が前記レジスタに関連するという前記判断ステップの判断に応答して、警告信号を送るステップと、

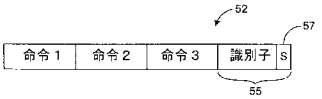
を含むコンピュータ・プログラム内のハザードを検出するスーパースカラー処理方法。

【００７８】
（７）前記命令のそれぞれが分析されたという判断に応答して、前記エントリをリセットするステップを含む上記（６）に記載のコンピュータ・プログラム内のハザードを検出するスーパースカラー処理方法。
【００７９】
（８）他の命令を分析して、前記他の命令が新しい命令グループに関連するかどうか判断するステップと、
前記他の命令が新しい命令グループに関連するという前記判断ステップの判断に応答して、前記エントリをリセットするステップと、
を含む上記（６）に記載のコンピュータ・プログラム内のハザードを検出するスーパースカラー処理方法。
【００８０】
（９）前記他の命令を分析するステップと、
前記他の命令が前記レジスタに関連するという前記判断ステップの判断に応答して、前記エントリにデータ値を記憶するステップと、
を含む上記（６）に記載のコンピュータ・プログラム内のハザードを検出するスーパースカラー処理方法。
【００８１】
（１０）前記他の命令が書き込み命令であるという判断に応答して、前記記憶するステップを実行する上記（９）に記載のコンピュータ・プログラム内のハザードを検出するスーパースカラー処理方法。
【００８２】
【発明の効果】
ある１つの命令グループが、同じ命令グループの他の命令とデータ・ハザードを規定する命令を含んでいるかどうかを判断することができる。
【図面の簡単な説明】
【図１】本発明に従う処理システムを使用するコンピュータ・システムを示すブロック図。
【図２】図１の処理システムに送られる命令の典型的なバンドルを示すブロック図。
【図３】少なくとも１つの命令グループを定義する命令バンドルの典型的なセットを示すブロック図。
【図４】図１の処理システムの詳細を示すブロック図。
【図５】図１の処理システムの処理段階を示すフローチャート。
【図６】図４の制御機構の好ましい実施形態のアーキテクチャおよび機能を示すフローチャート。
【図７】本発明の他の実施形態に従う、図４の制御機構によって使用されることのできる付加的なアーキテクチャおよび機能を示すフローチャート。
【符号の説明】
２０　処理システム
４１　コンピュータ・プログラム
７２　IDU(命令分散ユニット)
７５　パイプライン
８５　制御機構
９１　メモリ

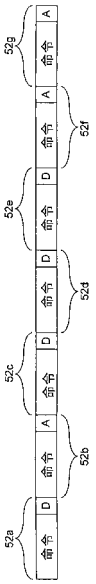
【図１】


15
44
メモリ
42
システム・マネージャ
41
プログラム
46
コンパイラ
20
処理システム
31
クロック
22
ローカル・インタフェース
25
入力装置
27
ディスプレイ
33
ネットワーク・インタフェース
32
ディスク
29
プリンタ
35
ネットワーク

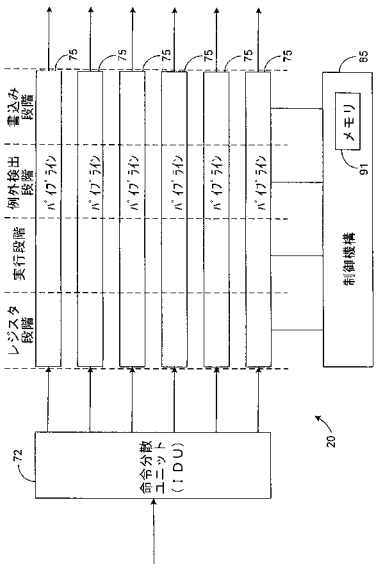
【図２】


52
57
命令１
命令２
命令３
識別子
S
55

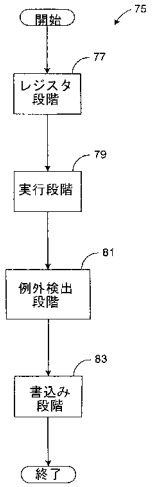
【図３】


52g
A
命令
52f
D
52e
52d
52c
52b
52a

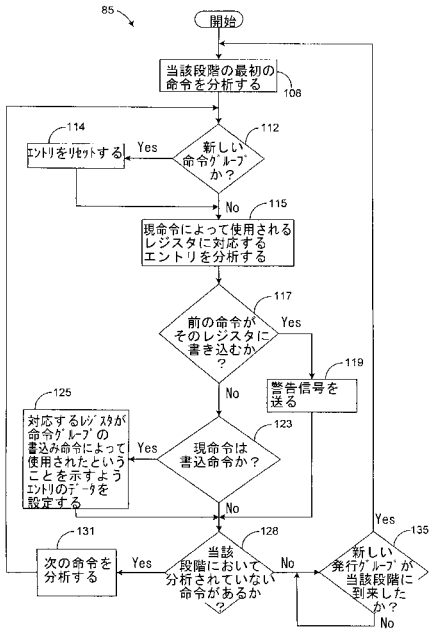
【図４】


75
書込み段階
例外検出段階
パイプライン
実行段階
レジスタ段階
85
メモリ
91
制御機構
20
72
命令分散ユニット（IDU）

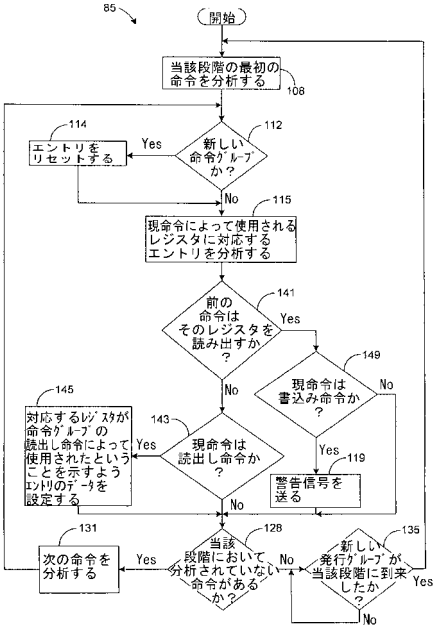
【図５】


75
開始
77
レジスタ段階
79
実行段階
81
例外検出段階
83
書込み段階
終了

【図６】


85
開始
当該段階の最初の命令を分析する
108
新しい命令ｸﾞﾙｰﾌﾟか？
112
Yes
エントリをﾘｾｯﾄする
114
No
現命令によって使用されるレジスタに対応するエントリを分析する
115
前の命令がそのレジスタに書き込むか？
117
Yes
警告信号を送る
119
No
現命令は書込命令か？
123
Yes
対応するﾚｼﾞｽﾀが命令ｸﾞﾙｰﾌﾟの書込み命令によって使用されたということを示すようｴﾝﾄﾘのﾃﾞｰﾀを設定する
125
No
当該段階において分析されていない命令があるか？
128
Yes
次の命令を分析する
131
No
新しい発行ｸﾞﾙｰﾌﾟが当該段階に到来したか？
135
Yes
No

【図７】


85
開始
当該段階の最初の命令を分析する
108
新しい命令ｸﾞﾙｰﾌﾟか？
112
Yes
エントリをリセットする
114
No
現命令によって使用されるレジスタに対応するエントリを分析する
115
前の命令はそのレジスタを読み出すか？
141
Yes
現命令は書込み命令か？
149
No
Yes
警告信号を送る
119
No
現命令は読出し命令か？
143
Yes
対応するﾚｼﾞｽﾀが命令ｸﾞﾙｰﾌﾟの読出し命令によって使用されたということを示すようｴﾝﾄﾘのﾃﾞｰﾀを設定する
145
No
当該段階において分析されていない命令があるか？
128
Yes
次の命令を分析する
131
No
新しい発行ｸﾞﾙｰﾌﾟが当該段階に到来したか？
135
Yes
No

フロントページの続き

(72)発明者　ロニー・リー・アーノルド
アメリカ合衆国８０５２８コロラド州フォート・コリンズ、スティルウォーター・クリーク・ドライブ　２２００

審査官　後藤　彰

(56)参考文献　特開平１０－２３２７７９（ＪＰ，Ａ）
特開平１０－１４３３６５（ＪＰ，Ａ）
特開平１０－９７４２４（ＪＰ，Ａ）
特開平２－１４８３２９（ＪＰ，Ａ）
特開昭５３－１０８２５４（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F　9/38